LucidLink

WHITEPAPER

# Methods and Best Practices for Accessing AWS S3 Cloud Object Storage

## Finding the right way to consume object storage

Organizations are creating more and more data. There are many studies covering this rate of increase, resulting in some pretty outrageous statistics, like one for example, that in the last two years we created 90% of the data that exists on earth[1]. Managing data growth is a problem all organizations face, and doing so in a way that is affordable and scalable means that more and more people look to object storage as a solution.

Growth in object storage consumption is far out-stripping that of traditional block or file based storage. Part of the reason stems from the ease of adopting cloud resources and the massive data creation that comes from DevOps, IoT and AI.

Object storage is chosen because of its attractiveness as a low-cost, elastic and highly durable solution, it's great for static data, backup and archive as well as cloud native applications. For this reason cloud storage is continuing to grow at a rapid pace and shows no signs of slowing down any time soon, so a common question from AWS clients is, *"How can I mount and use S3 storage for production workloads?"*

We know object storage makes a lot of sense. But what is the best way to consume it for your specific use case? A cursory search will show plenty of products and solutions available. However, when peeling the marketing layers off these solutions, it soon becomes clear that they come with significant trade-offs in the areas of performance, flexibility and security.

### Consuming object storage securely

Security should not get in the way of performance, flexibility and overall usability. It is challenging to get this combination right. Many clients in a variety of markets are justifiably concerned about safeguarding data and meeting compliance rules. In order for cloud consumption to be a viable option there needs to be adequate security in place to ensure data is safe.

There are problems with the approach taken by the providers of many cloud based software-defined storage and cloud file systems today. They provide proprietary file sharing services using decades-old protocols. These solutions attempt to deliver these services with the same sovereignty requirements as on-premises counterparts.

Cloud-native file services require additional security considerations when it comes to their provisioning and use, in the cloud, from edge-to-cloud and cloud-to-cloud. You want to make sure a solution allows for bringing your own security with you, allowing you to meet your required compliance standards, where the secret keys used to encrypt data are the ones you own. It is important to consider the level of encryption used (such as AES-256) and that this is used at-rest and in-flight.

[1] https://www.forbes.com/sites/bernardmarr/2018/05/21/how-much-data-do-we-create-every-day-the-mind-blowing-stats-everyone-should-read/

## Overall Useability

There is no file system associated with object storage, only API access. A "PUT" or "GET" is used to store or retrieve data from an object storage bucket. Most file based applications are developed with local disk or local network type latencies in mind and do not perform well over distance. Therefore any use case for cloud object storage outside of backup and archive focuses on pulling files from the cloud to the device where the application resides.

This means that applications have to be API-aware, and made to ignore latency, which is fine when the application is natively written to work with object storage, otherwise this requires refactoring. In the case of a cloud-native application, sharing this object storage data with traditional applications and or end-users can be challenging. Let's look at the currently available methodologies.

### Cloud storage as a mount point

Several vendors like Expandrive and Mountain Duck abstract object storage API calls. Exposing object storage "PUT" and "GET" commands as a local operating system mount point. This provides an experience that appears local, but when requesting a file, this must fully download before being accessible. After changes are made these must be fully uploaded before use.

This is convenient for casual access, and small datasets, but not great for production, application access or collaborative use due to the lack of performance and the lack of a true native file system experience. So besides adding the appearance of convenience, these services do no thing to change the underlying issues with object storage.

### Full-sync or sync-on-demand

Cloud file synchronization was the first way to address object storage access, made popular over ten years ago by companies like Box, Dropbox and others. A 'big hammer' approach, the idea is simple, put a copy of the file wherever you may need it and keep changes synchronized across all devices. While the approach is acceptable for individual users with small files and small datasets, it only addresses casual file sharing requirements.

As desired use cases change, and datasets and file sizes grow this approach does not scale. It introduces significant performance issues for large data sets. One cannot possibly expect multiple copies of application or file data to exist for each individual connected entity, whether this is a containerized application instance consuming data from a central share or a user accessing a document from a shared team workspace.

Cloud file synchronization solutions lack control and security as to how accessed data is stored locally as well as in the cloud, meaning you have to put complete trust both in your individual users accessing the data as well as the provider of the synchronization service.

While accessing object storage through a file synchronization service is possible, these products simply are not designed to scale-up to be used as a primary data source in live production environments.

### Gateways and caching devices

Gateways like Nasuni, Panzura, Ctera and Avere can be used to allow client endpoints to access cloud object storage. These can either be physical or virtual appliances and vary in complexity, requiring initial configuration, day-to-day management and monitoring.

To allow access, endpoints need to be on the same local network as the gateway device, potentially requiring multiple gateway devices for multiple sites, increasing cost and complexity. In effect, users are trading storage devices for caching devices optimized to access object storage.

Besides the additional infrastructure required, there are scale-out and security ramifications. These gateways use the same protocols as traditional network file shares. If a gateway is involved, data transfer typically uses NFS or SMB protocols and data from all endpoint connections follows a single path, turning this gateway into the bottleneck.

## Cloud file storage systems

Cloud file storage systems like Amazon EFS and Azure Files allow shared data access for cloud workloads over NFS or SMB. Unfortunately these services are constrained to certain regions and standard network protocols. Similar to a standard file server endpoint, throughput is constrained by the bandwidth available on the file server side and has to be shared with all connected client endpoints. External file system access outside of the hosting cloud provider requires VPN access.

As proprietary cloud provider offerings, cloud file storage systems limit flexibility in storage choice. Depending on the storage used and the use case the difference in cost can be considerable. Especially when compared to object storage that offers tiering with different costs for active and archive data, this may not be justifiable.

While network shares and cloud file storage solves a problem, network protocols are not designed for high-latency links and are relatively inefficient when it comes to data transport. Traditional network shares are challenging to secure. There is quite a difference between connecting to a network share, and accessing data from a local mount-point with secure caching enabled that allows your object storage to behave like it is local disk.

## Cloud-native unified file service for object storage

The bottom line is that simply providing access to object storage via operating system mount points, sync-and-share, gateway solutions, or network shares is not enough. Cloud object storage consumption requires a more sophisticated approach.

A better option is a cloud-native solution, designed to be a distributed file service that is built on top of S3-compatible object storage. Clients access files in-place via a local mount point even though the actual data may be in-cloud. Data is encrypted end-to-end and keys are user managed, where the provider service has no visibility of the data or the metadata content.

To deal with latency, these solutions distribute metadata and present files as if they are local, prefetching and caching only the blocks the application requires, streaming on-demand. This reduces chatter between applications and the file system. Client systems query their local metadata copy and make direct calls on reads when specific data is required from object storage, which is cached, reducing egress cost. For writes data is cached before being offloaded to object storage.

This approach works because individual files are split into blocks of multiple objects. Instead of being a one to one relationship, a proprietary data layout is used. This means that for very large files multiple connections can be opened to improve data access performance and allows random access to modify only parts of the file being changed.

The benefit to this model is that you retain full local visibility of all your data, without consuming local storage. This provides virtually instant access to your files and allows seamless integration into existing client workflows and applications.

## In Conclusion

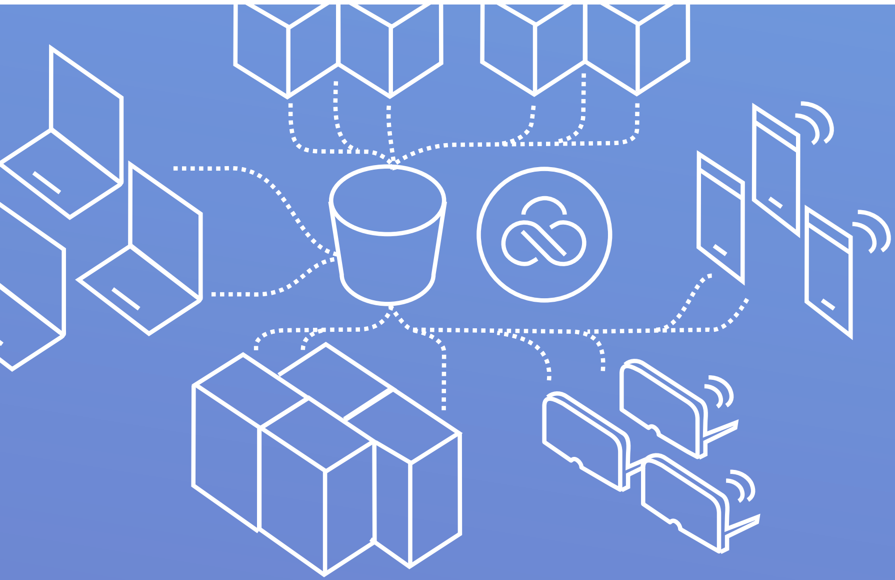
As discussed, there are many ways of accessing cloud object storage. In order to find the one that ultimately delivers the correct balance of security, performance, flexibility and cost, a proper assessment of each solution should include the following factors:

- The number of locations needing to access the data
- The amount of users or applications connected to the same storage
- How often and quickly files need to be accessed
- The size and types of files being stored
- The security and compliance requirements for this data
- The amount of control and manageability required

Determining the relative importance of these factors, given budget constraints will help companies in determining the correct solution for their specific requirements. We think LucidLink, which addresses all areas at a lower cost, is a great starting point.

**Learn more**
**lucidlink.com**

**Sign up for free trial**
**lucidlink.com/webportal/register**



LucidLink | Stream your data from any cloud